REMARKS

New claims 48-53 are added. These claims recite no impermissible new matter.

Thus, claims 27-31 and 33-53 are the claims now pending in this Application.

Claims 27 and 47 are amended so as more clearly to recite patentable features of the invention.

Claims 27-47 are rejected under 35 U.S.C. §102(b) over Wold, U.S. Patent No. 5,724,589. This rejection is traversed.

In an objected-oriented programming paradigm, software objects contain data and methods that request the services of other objects or when invoked perform services for other objects. Among the problems recognized and solved by Applicant's claimed invention is that of having a group of software units which need not know which other objects and which other methods to call or request services from. (This discussion merely illustrates aspects of Applicant's invention, however any particular embodiment of Applicant's invention does not necessarily perform or embody the features discussed herein.) According to an aspect of Applicant's claimed invention, an executive software unit manages links between the software units of an ensemble of units and between the software units and the outside of the ensemble. Applicant respectfully submits that Wold does not disclose or suggest this problem, let alone disclose or suggest the solutions provided by Applicant's claimed invention.

Independent claim 27 requires, *inter alia*, a software ensemble comprising an executive software unit containing links between each software unit of the plurality of

software unit including a method and data. Further, independent claim 47 requires, *inter alia*, a software ensemble comprising a linkage in an executive software unit that identifies the one or more of the plurality of software units that is to receive the message.

Wold discloses a software development system with a software unit property-method programming model. (Wold, Abstract.) Wold discloses a software development system in the C++ programming language, in which a change in the value of a property of an object (for example, the color property of a scroll bar object) will be an event that invokes an action by a method in a second object. Wold describes a C++ implementation of the "Closure" concept, as the binding a function pointer ("where" to call) with an object pointer ("who" is being called) (Wold, col. 4, lines 8-19). A "Closure" is limited to one object and one method and it does not allow finding the one or more of the methods to be invoked at the one or more of the plurality of software units when a message is sent to the output variable of the software unit as for example recited in independent Claim 47. Since absolute addresses are used in the definition of a "Closure," a "Closure" does not disclose or suggest a mechanism to implement software units such as software objects or connections.

Further, Wold discloses an "event handler" object that includes a dispatcher method and an event handler method, such that the dispatcher method is invoked when an event source (for example the scroll bar object) invokes the function performed by the event handler. That is, Wold discloses a dispatcher linking one object (event source) with one action (the method event handler in the event sink object, also known as the event handler object). (Wold, Fig. 5; col. 11, lines 1-14.)

Wold does not disclose or suggest a software ensemble comprising an executive software unit containing links between each software unit of the plurality of software units and between software units and an outside of the software ensemble. Further, Wold does not disclose or suggest a software ensemble comprising a linkage in an executive software unit that identifies the one or more of the plurality of software units that is to receive the message. First, Wold does not disclose or suggest a software ensemble comprising an executive software unit and a plurality of software units. As discussed, Wold discloses a dispatcher method and an event handler method included in an event sink object, and a second object, the event source object, a change in the property of which invokes the event handler method by means of the dispatcher method of the event sink object. (Reference Numeral 510 of Fig. 5 of Wold shows the event sink object, Reference Numeral 517 of the event sink object being the dispatcher method and "virtual member function" being the event handler method, and Reference Numeral 530 shows the event source object).

Further, Wold does not disclose or suggest a software ensemble comprising an executive software unit that links the plurality of software units with each other and with software units outside of the software ensemble. Therefore, Wold does not disclose or suggest Applicant's invention as claimed in independent claims 27 and 47.

Claims 28-46 depend from independent claim 27 and thus incorporate novel and nonobvious features thereof. Therefore, claims 28-46 are patentably distinguishable over the prior art for at least the reasons that claim 27 is patentably distinguishable over the prior art.

New claims 48-53 are added so as more fully to claim patentable aspects of Applicant's claimed invention. Claims 48-53 depend from independent claims 27 and 47 and are thus patentably distinguishable over the prior art for at least the reasons that claims 27 and 47 are patentably distinguishable over the prior art.

Filed herewith is a Request for Continued Examination (RCE), with fee, and a Petition for Extension of Time for a one-month extension, with fee.

Respectfully submitted

George Brieger
Registration No. 52,652

Scully, Scott, Murphy & Presser
400 Garden City Plaza
Garden City, New York 11530
(516) 742-4343

GB:eg